



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/523,877	03/13/2000	Peter Warnes	ARC.005A	6131

27299 7590 05/29/2003

GAZDZINSKI & ASSOCIATES
11440 WEST BERNARDO COURT, SUITE 375
SAN DIEGO, CA 92127

EXAMINER

HUISMAN, DAVID J

ART UNIT	PAPER NUMBER
----------	--------------

2183

DATE MAILED: 05/29/2003

13

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No.

09/523,877

Applicant(s)

WARNES ET AL. *Se*

Examiner

David J. Huisman

Art Unit

2183

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 25 March 2003.
- 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-5, 14-20, 23 and 25-43 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-5, 14-20, 23 and 25-43 is/are rejected.
- 7) ☒ Claim(s) 17 is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 13 March 2000 is/are: a) ☒ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
- 11) ☐ The proposed drawing correction filed on _____ is: a) ☐ approved b) ☐ disapproved by the Examiner.
If approved, corrected drawings are required in reply to this Office action.
- 12) ☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

- 13) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. _____.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
* See the attached detailed Office action for a list of the certified copies not received.
- 14) ☒ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).
a) ☐ The translation of the foreign language provisional application has been received.
- 15) ☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

- 1) ☐ Notice of References Cited (PTO-892) 4) ☐ Interview Summary (PTO-413) Paper No(s) _____
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948) 5) ☐ Notice of Informal Patent Application (PTO-152)
- 3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ 6) ☐ Other: _____

Art Unit: 2183

DETAILED ACTION

1. Claims 1-5, 14-20, 23, and 25-43 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: #12. Amendment 'B' as received on 3/25/2003.

Claim Objections

3. Claim 17 is objected to because of the following informalities: The use of the word "one" with "values" does not seem appropriate. The phrase should be reworded so that the words agree with each other. Appropriate correction is required.

Response to Arguments

4. Applicant's arguments filed on March 25, 2003, have been fully considered but they are not persuasive.
5. In the remarks, the Applicant argues the novelty of amended claim 1, 14, 17, and 23 on pages 9-10, in substance that:

"None of the cited references including Lee seemingly teach or suggest providing at least one user-configurable mode (e.g., delay slot mode) which can be used for branch/jump control or otherwise."

This argument is not found persuasive for the following reason:

It should be noted that the branch instructions taught by Lee would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the

Art Unit: 2183

programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction.

6. In the remarks, the Applicant argues the novelty of amended claim 20 on page 10, in substance that:

“Lee teaches or suggests no fourth discrete mode, let alone a fourth mode which allows selection of different number of stall cycles from another mode.”

This argument is not found persuasive for the following reason:

For a response to this argument, the applicant is directed to view the rejections of claims 20 and 34 below. From these rejections, it will become evident that Lee has taught a fourth discrete mode and a mode that may have at least two stall cycles.

Claim Rejections - 35 USC § 102

7. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

8. Claims 1-5, 14-18, 20, 23, 25, 29, and 33-43 are rejected under 35 U.S.C. 102(b) as being anticipated by Lee et al., U.S. Patent No. 4,755,966 (as applied in the previous Office Action and herein referred to as Lee).

Art Unit: 2183

9. Referring to claim 1, Lee has taught a method of controlling the execution of instructions within a pipelined processor, comprising:

a) providing an instruction set comprising a plurality of instruction words. Lee discloses in column 3, lines 46-47, that each instruction within the instruction set contains a 6-bit opcode, which means a total of 64 instructions could exist within the system. Lee also discloses in column 6, lines 36-39, that the system contains a floating-point unit, which means floating-point instructions would exist.

b) each of said instruction words comprising a plurality of data bits. See column 3, lines 40-42. Each instruction is 32 bits.

c) at least one of said words comprising a jump instruction having at least one user-configurable mode associated therewith. Fig.2, component 102, shows the use of a branch (jump) instruction. Also, it should be noted that these branch instructions would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction.

d) assigning one of a plurality of values to at least one of said data bits of said at least one jump instruction. See column 3, lines 46-51. Lee discloses that each branch instruction contains a nullify bit and a displacement sign bit. These bits can be assigned a value (0 or 1) as shown in Fig.3.

Art Unit: 2183

e) controlling the execution of at least one subsequent instruction within said pipeline based on said one assigned value of said at least one data bit when said at least one jump instruction is decoded. See column 3, lines 58-61. By setting or clearing this nullify bit, a subsequent instruction's execution is controlled.

10. Referring to claim 2, Lee has taught a method as described in claim 1. Lee has further taught that the act of assigning comprises identifying a plurality of data bits within said at least one jump instruction and assigning one of two discrete values to each of said data bits, the combination of said two discrete values representing at least three jump delay slot modes within said processor. See column 5, lines 7-46 for a description of the 5 different jump delay slot modes shown in Fig. 2. In essence, the nullify bit (Fig. 1, component 507) and the displacement sign bit (Fig. 1, component 508) are checked by the system and the combination of those values will determine the jump delay slot mode. It is inherent that each bit would be assigned one of two discrete values (i.e. 0 or 1) since a processor only understands binary values.

11. Referring to claim 3, Lee has taught a method as described in claim 2. Lee has further taught that the act of controlling the execution based on said discrete values comprises selecting at least one mode from the group comprising:

a) executing said at least one subsequent instruction under all circumstances. See column 5, lines 50-53, and Fig. 3.

b) executing said at least one subsequent instruction only if a jump occurs. See column 5, lines 53-57, and Fig. 3. The delay slot instruction will be executed when a jump occurs and the displacement is negative. The delay slot instruction will not be executed if a jump does not occur and the displacement is negative.

Art Unit: 2183

c) stalling the pipeline or inserting a bubble into the pipeline if a jump occurs. See column 5, lines 32-37, and Fig.2. Note that if a jump occurs and the displacement is positive (as shown in Fig.2, component 112), the delay slot instruction would be fetched but not executed. Therefore, in order to kill the unwanted instruction, the pipeline would be stalled for at least a single cycle.

12. Referring to claim 4, Lee has taught a method as described in claim 3. Lee has further taught that at least one jump instruction comprises a conditional branch instruction. See column 2, lines 62-64, and note the conditional component 202.

13. Referring to claim 5, Lee has taught a method as described in claim 1. Furthermore, note that the displacement sign bit is assigned one value as is the nullify bit. Therefore, claim 5 is rejected for the same reasons set forth in the rejection of claim 3 above.

14. Referring to claim 14, Lee has taught a digital processor comprising:

a) a processor core having a multistage instruction pipeline, said core being adapted to decode and execute an instruction set comprising a plurality of instruction words. Fig.5 shows a 4-stage pipeline that is further described in column 6, line 56, to column 7, line 39. Furthermore, it is inherent that the processor will decode and execute multiple instructions (as established in the rejection of claim 1) from an instruction set.

b) a data interface between said processor core and an information storage device. It is inherent that in order for a processor to execute instructions, they must be stored in the processor's memory. Therefore, the instructions would be stored in an information storage device that is directly accessible by the processor.

c) an instruction set comprising a plurality of instruction words, at least one of said instruction words being a user-configurable jump instruction containing data defining a plurality of jump

Art Unit: 2183

delay slot modes, said plurality of modes controlling the execution of instructions within said instruction pipeline of said processor core in response to said at least one jump instruction word within said instruction set. This portion of claim 14 is rejected for the same reasons set forth in the rejection of claim 3 above. Also, as discussed in the rejection of claim 1, it should be noted that these branch instructions would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction.

15. Referring to claim 15, Lee has taught a digital processor as described in claim 14.

Furthermore, claim 15 is rejected for the same reasons set forth in the rejection of claim 3 above.

16. Referring to claim 16, Lee has taught a digital processor as described in claim 14. Lee has further taught that at least one jump instruction comprises a conditional branch instruction having an associated logical condition, the execution of a jump to the address within said information storage device specified by said at least one conditional branch instruction being determined by said logical condition. See Fig.3, component 202, and note that the branch will be taken or not taken based on some condition. Furthermore, it is inherent that the execution of the branch will cause a jump to the address (specified by the instruction) within the information storage device.

Art Unit: 2183

17. Referring to claim 17, Lee has taught a digital processor having at least one pipeline and an associated data storage device, wherein the execution of instructions within said at least one pipeline is controlled by the method comprising:

a) storing an instruction set within said data storage device, said instruction set comprising a plurality of instruction words, each of said instruction words comprising a plurality of data bits, at least one of said instruction words comprising a user-configurable branch instruction directing branching to a first address within said data storage device. As discussed above in the rejections of claim 14, Lee has taught an instruction set with a plurality of instructions that would inherently comprise a plurality of bits and would inherently be stored in a data storage device in order to processor-accessible and executable. Furthermore, from the rejection of claim 16, it is inherent that a branch will cause a jump to a specified address within the data storage device. Finally, it should be noted that these branch instructions would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the programmer/user must specify the target address of the branch. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction.

b) assigning one of a plurality of values to each of said data bits of said at least one branch instruction. See column 3, lines 46-51. Lee discloses that each instruction contains 32 data bits of information. This information includes source registers, displacements, an opcode, a condition fields, and a nullify bit and a displacement sign bit. Each of these bits is assigned a value (0 or 1). For instance, if the user decides to use a branch instruction, then he/she must assign the instruction the appropriate opcode, along with the target address (displacement),

Art Unit: 2183

condition, etc. In essence, each of the instruction's data bits must be set accordingly in order for the system to recognize that it's a branch instruction.

c) decoding said at least one branch instruction including said one values. Since, the values of the nullify bit and displacement sign bit are part of each branch instruction, it follows that the value will be decoded as the branch instruction is decoded. See Fig. 1 and column 3, lines 46-51.

d) determining whether to execute an instruction within said pipeline in a stage preceding that of said at least one branch instruction based at least in part on said one values. The delay slot instruction would be in a pipeline stage preceding that of the branch instruction. And, Lee discloses a system in which the execution of the delay slot instruction is determined based on the value of the nullify bit and/or displacement bit.

e) branching to said first address based on said at least one branching instruction. Recall from claim 16, that it is the inherent nature of a branch instruction (when taken) to jump to a specified address.

18. Referring to claim 18, Lee has taught a processor as described in claim 17. Furthermore, it is inherent that the data bits comprise binary data. A processor can only recognize and understand zeroes and ones.

19. Referring to claim 20, Lee has taught a method of controlling program operation of a multi-stage pipelined digital processor, comprising:

a) storing an instruction set within said data storage device, said instruction set comprising a plurality of instruction words, each of said instruction words comprising a plurality of data bits, at least one of said instruction words comprising a branch instruction directing branching to a first address within said data storage device based on a first parameter. This portion of claim 20

Art Unit: 2183

is rejected for the same reasons set forth in the rejection of claim 17 above. Furthermore, the first parameter could be considered the logic that determines whether the branch is taken or not taken (ex. a flag is checked and compared to a value).

b) defining a plurality of jump delay slot modes comprising:

(i) executing a subsequent instruction under all circumstances. See column 5, lines 50-53, and Fig.3.

(ii) executing a subsequent instruction only if jumping occurs. See column 5, lines 53-57, and Fig.3. The delay slot instruction will be executed when a jump occurs and the displacement is negative. The delay slot instruction will not be executed if a jump does not occur and the displacement is negative.

(iii) stalling the pipeline for one cycle if jumping occurs. See column 5, lines 32-37, and Fig.2. Note that if a jump occurs and the displacement is positive (as shown in Fig.2, component 112), the delay slot instruction would be fetched but not executed. Therefore, in order to kill the unwanted instruction, the pipeline would be stalled for at least a single cycle.

(iv) stalling the pipeline for two or more cycles if jumping occurs. See Fig.5, column 5, lines 58-63, and column 7, lines 21-25. Note that the pipeline includes four stages: an address generation stage (A), a fetch stage (F), an execution stage (E), and a write stage (W). It has been disclosed that the target address of the branch is not determined until the end of the execution stage. By this time, the delay slot instruction will have been fetched, and the predicted target address will be supplied to the program counter for fetching in the next cycle. See column 7, lines 6-10. If the delay slot instruction ends up being

Art Unit: 2183

nullified (based on the displacement and nullify bits) and the predicted target is incorrect, then both instructions will need to be cancelled, resulting in a 2-cycle stall.

c) assigning at least one of said plurality of jump modes to at least two of said data bits of said at least one branch instruction. Note that both the displacement bit and nullify bit control the jump mode. These bits make up two of the instruction's data bits.

d) decoding said at least one branch instruction including said at least two data bits. Note that when the branch instruction is decoded, the nullify and displacement bits are decoded as well in order to determine the mode. See Fig.3.

e) controlling said pipeline based at least in part on said at least two data bits and said first parameter. Again, the first parameter would involve the condition field of the branch instruction. See column 3, line 49. For instance, this parameter might specify to perform a jump if a certain value is greater-than or equal to 0. In addition, the nullify and displacement bits play a role in controlling the operation of the pipeline. See Fig.3.

20. Referring to claim 23, Lee has taught a digital processor comprising:

a) processing means having a multistage data pipeline, said processing means being adapted to decode and execute an instruction set comprising a plurality of instruction words. Recall that Lee has disclosed a multi-stage pipeline in Fig.5 and column 6, lines 59-66. Also the decoding and executing is disclosed in column 7, lines 12-19. Finally, it has been established in the rejection of claim 1 above that Lee has taught an instruction set with a plurality of instruction words.

Art Unit: 2183

b) means for storing data. It is inherent that the processor must have a memory from which instructions can be read. Also, Lee has taught that the system includes a register file for temporary storage. See column 6, lines 36-40.

c) data interface means for transferring data between said processing means and said means for storing data. In order for instructions to be processed, the processor must fetch them from memory first. Therefore, it is inherent that an interface exists between the processor and the memory.

d) an instruction set comprising a plurality of instruction words, at least one of said instruction words being a user-configurable jump instruction containing data defining a plurality of jump control means, said plurality of jump control means controlling the execution of instructions within said data pipeline of said processing means in response to said at least one jump instruction word within said instruction set. This portion of claim 23 is rejected for the same reasons set forth in the rejection of claim 14 above. Also, as discussed in the rejection of claim 1, it should be noted that these branch instructions would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction.

21. Referring to claims 25 and 29, Lee has taught a method and digital processor as described in claims 1 and 14, respectively. Lee has further taught that at least one of said plurality of instruction words comprises an op-code and a plurality of fields, each of said fields comprising a

Art Unit: 2183

plurality of bits (see column 3, lines 46-51), said at least one instruction word being encoded according to the method comprising:

- a) associating a first of said fields with a first data source. See column 3, lines 47-48 (component 503).
- b) associating a second of said fields with a second data source. See column 3, lines 48-49 (component 504).
- c) performing a logical operation using said first and second data sources as operands, said logical operation being specified by said op-code. See column 3, lines 51-55. In this case, the opcode specifies a compare and branch instruction where the comparison is performed between the contents of the two specified registers.

22. Referring to claim 33, Lee has taught a method of controlling the execution of instructions within a user-configured pipelined processor, comprising:

- a) providing an instruction set comprising a plurality of instruction words, each of said instruction words comprising a plurality of data bits, at least one of said words comprising a branch instruction having at least one user-configurable mode and a plurality of other modes adapted for controlling the execution of at least one subsequent instruction. Lee discloses in column 3, lines 40-47, that each instruction (comprising 32 data bits) within the instruction set contains a 6-bit opcode, which means a total of 64 instructions could exist within the system. In addition, Fig.2, component 102, shows the use of a branch (jump) instruction. It should be noted that these branch instructions would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In

Art Unit: 2183

addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction.

b) assigning one of a plurality of values to at least one of said data bits of said at least one branch instruction. See column 3, lines 46-51. Lee discloses that each branch instruction contains a nullify bit and a displacement sign bit. These bits can be assigned a value (0 or 1) as shown in Fig.3.

c) controlling the execution of at least one subsequent instruction within said pipeline based on said one assigned value of said at least one data bit when said at least one branch instruction is decoded. See column 3, lines 58-61. By setting or clearing this nullify bit, a subsequent instruction's execution is controlled.

23. Referring to claim 34, Lee has taught a method of controlling the execution of instructions within a pipelined processor, comprising:

a) providing an instruction set comprising a plurality of instruction words, each of said instruction words comprising a plurality of data bits, at least one of said words comprising a jump instruction. Lee discloses in column 3, lines 40-47, that each instruction (comprising 32 data bits) within the instruction set contains a 6-bit opcode, which means a total of 64 instructions could exist within the system. In addition, Fig.2, component 102, shows the use of a branch (jump) instruction.

b) assigning one of a plurality of values to first and second of said data bits of said at least one jump instruction, said first and second bits adapted to define four discrete jump modes. By

Art Unit: 2183

setting the nullify and displacement bits, eight jump modes can actually be achieved. The modes are as follows:

<i>Nullify Bit Value</i>	<i>Displacement Bit</i>	<i>Branch Outcome</i>	<i>Branch Mode</i>
0	0	Taken	Jump Forward w/ Delay Slot Execution
0	0	Not Taken	Delay Slot Execution
0	1	Taken	Jump Backward w/ Delay Slot Execution
0	1	Not Taken	Delay Slot Execution
1	0	Taken	Jump Forward w/o Delay Slot Execution
1	0	Not Taken	Jump Forward w/ Delay Slot Execution
1	1	Taken	Jump Backward w/ Delay Slot Execution
1	1	Not Taken	Jump Backward w/o Delay Slot Execution

From the above modes, it can be recognized that there are at least five discrete modes:

- 1) no jumping with delay slot execution
- 2) jumping forward with delay slot execution
- 3) jumping forward without delay slot execution
- 4) jumping backward with delay slot execution
- 5) jumping backward without delay slot execution

Given that a branch is always either taken or not taken, the branch mode is then dictated by the nullify and displacement bits. For instance, assume that a first branch instruction in a program is encountered twice (taken the first time and not taken the second time). If the nullify and displacement bits for this instruction are 11, then when the branch is taken, the bits will cause the delay slot instruction to be executed, while when the branch is not taken, the bits will cause the delay slot instruction to be nullified.

Art Unit: 2183

c) controlling the execution of at least one subsequent instruction within said pipeline based on said assigned values of said first and second data bits when said at least one jump instruction is decoded. See column 3, lines 58-61. By setting or clearing this nullify bit, a subsequent instruction's execution is controlled.

24. Referring to claim 35, Lee has taught a method of controlling the execution of instructions within a pipelined processor, comprising:

a) providing an instruction set comprising a plurality of instruction words, each of said instruction words comprising a plurality of data bits, at least one of said words comprising a jump instruction. Lee discloses in column 3, lines 40-47, that each instruction (comprising 32 data bits) within the instruction set contains a 6-bit opcode, which means a total of 64 instructions could exist within the system. In addition, Fig. 2, component 102, shows the use of a branch (jump) instruction.

b) assigning one of a plurality of values to first and second of said data bits of said at least one jump instruction, said first and second bits adapted to define four discrete jump modes, said four discrete jump modes including one user-defined jump mode. This portion of claim 35 is rejected for the same reasons set forth in the rejection of claim 34(b) above. Also, it should be noted that these branch instructions would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction.

Art Unit: 2183

c) controlling the execution of at least one subsequent instruction within said pipeline based on said assigned values of said first and second data bits when said at least one jump instruction is decoded. See column 3, lines 58-61. By setting or clearing this nullify bit, a subsequent instruction's execution is controlled.

25. Referring to claim 36, Lee has taught a digital processor having:

a) at least one pipeline and an associated data storage device containing at least a portion of an instruction set comprising a plurality of instruction words, each of said instruction words comprising a plurality of data bits, at least one of said words comprising a jump instruction.

Firstly, it is inherent that in order for a processor to execute instructions, they must be stored in the processor's memory (data storage device). Furthermore, Lee discloses in column 3, lines 40-47, that each instruction (comprising 32 data bits) within the instruction set contains a 6-bit opcode, which means a total of 64 instructions could exist within the system. In addition, Fig. 2, component 102, shows the use of a branch (jump) instruction.

b) wherein the execution of instructions within said at least one pipeline is controlled by:

(i) the assignment of one of a plurality of values to first and second of said data bits of said at least one jump instruction, said first and second bits adapted to define four discrete jump modes. This portion of claim 36 is rejected for the same reasons set forth in the rejection of claim 34(b) above.

(ii) the execution of at least one subsequent instruction within said pipeline based on said assigned values of said first and second data bits, when said at least one jump instruction is decoded. See column 3, lines 58-61. By setting or clearing this nullify bit, a subsequent instruction's execution is controlled.

Art Unit: 2183

26. Referring to claim 37, Lee has taught an extensible pipelined digital processor having an instruction set comprising a plurality of basecase instructions and at least one extension instruction, at least one of said basecase and extension instructions comprising a branch instruction having at least one user-configurable mode and a plurality of other modes controlling the execution of at least one instruction in a delay slot following said branch instruction within said pipeline. Note that Lee's processor would include basecase instructions, which could be interpreted as being the regular instructions that can be executed on the processor, such as loads, stores, branches, arithmetic, logical, and floating-point instructions. See column 6, lines 36-39, and note that the system contains a floating-point unit, which means floating-point instructions would exist. Extension instructions can be interpreted as the branch instructions which utilize the extra bits within the instruction word, i.e., the nullify and displacement bits. It should be noted that these branch instructions would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction. Furthermore, multiple delay slot modes can be selected based on the values of these bits. See Fig. 3. Finally, see column 3, lines 58-61. By setting or clearing this nullify bit, a subsequent instruction's execution is controlled.

27. Referring to claim 38, Lee has taught an extensible pipelined digital processor having an instruction set comprising a plurality of basecase instructions and at least one extension instruction, at least one of said basecase and extension instructions comprising a branch

Art Unit: 2183

instruction including two data bits defining four discrete modes controlling the execution of at least one instruction in a delay slot following said branch instruction within said pipeline. Note that Lee's processor would include basecase instructions, which could be interpreted as being the regular instructions that can be executed on the processor, such as loads, stores, branches, arithmetic, logical, and floating-point instructions. See column 6, lines 36-39, and note that the system contains a floating-point unit, which means floating-point instructions would exist.

Extension instructions can be interpreted as the branch instructions which utilize the extra bits within the instruction word, i.e., the nullify and displacement bits. These two bits define at least four delay slot modes as shown in the rejection of claim 34(b) above. Finally, see column 3, lines 58-61. By setting or clearing this nullify bit, a subsequent instruction's execution is controlled.

28. Referring to claim 39, Lee has taught an extensible pipelined digital processor having an instruction set, said processor comprising:

a) a basecase processor core configuration including a base instruction set. The basecase instructions could be interpreted as being the regular instructions that can be executed on the processor, such as loads, stores, branches, arithmetic, logical, and floating-point instructions.

b) and at least one user-configured extension instruction within said instruction set, said at least one extension instruction comprising a branch instruction having at least one user-defined mode and a plurality of other modes controlling the execution of at least one instruction in a delay slot following said branch instruction within said pipeline. Extension instructions can be interpreted as the branch instructions which utilize the extra bits within the instruction word, i.e., the nullify and displacement bits. It should be noted that these branch instructions would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the

Art Unit: 2183

programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction. Furthermore, multiple delay slot modes can be selected based on the values of these bits. See Fig.3. Finally, see column 3, lines 58-61. By setting or clearing this nullify bit, a subsequent instruction's execution is controlled.

29. Referring to claim 40, Lee has taught an extensible pipelined digital processor having basecase and extension instruction sets, at least one instruction within said basecase set comprising a branch instruction having at least four discrete modes for controlling the execution of at least one instruction in a delay slot following said branch instruction within said pipeline, said processor comprising:

- a) a basecase processor core configuration including said base instruction set.
- b) at least one user-customized extension instruction within said instruction set.

Note that Lee's processor would include basecase instructions, which could be interpreted as being the regular instructions that can be executed on the processor, such as loads, stores, branches, arithmetic, logical, and floating-point instructions. See column 6, lines 36-39, and note that the system contains a floating-point unit, which means floating-point instructions would exist. Extension instructions can be interpreted as the branch instructions which utilize the extra bits within the instruction word, i.e., the nullify and displacement bits. These extension instructions can also be thought of as a subset of the basecase instructions, in that they are still executed by the basecase processor core. However, what sets them apart from the rest is the utilization of the two extra bits. It should be noted that these branch instructions would have at

Art Unit: 2183

least one user-configurable mode in that for every branch instruction, regardless of the mode, the programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction. Furthermore, multiple delay slot modes can be selected based on the values of these bits. See Fig.3. Finally, see column 3, lines 58-61. By setting or clearing this nullify bit, a subsequent instruction's execution is controlled.

30. Referring to claim 41, Lee has taught a digital processor having at least one pipeline and an associated data storage device containing at least a portion of an instruction set comprising a plurality of instruction words, each of said instruction words comprising a plurality of data bits, at least one of said words comprising a branch instruction. Firstly, it is inherent that in order for a processor to execute instructions, they must be stored in the processor's memory (data storage device). Furthermore, Lee discloses in column 3, lines 40-47, that each instruction (comprising 32 data bits) within the instruction set contains a 6-bit opcode, which means a total of 64 instructions could exist within the system. In addition, Fig.2, component 102, shows the use of a branch (jump) instruction. Lee has further taught that the execution of instructions within said at least one pipeline is controlled by:

- (i) the assignment of one of a plurality of values to at least first, second, and third of said data bits of said at least one branch instruction, said first, second, and third bits adapted to define at least four discrete branch modes. It should be noted that the nullify bit and the displacement bit values are set and they contribute to the selection of one of at least four of the possible selected jump modes, as shown in the table in the rejection of claim 34(b)

above. Furthermore, whether the branch is taken or not depends on the condition provided in the condition field of the branch instruction. By setting the bits in the condition field to unique values, the condition to be determined will change. For instance, the condition may result in the branch being taken if some variable is greater-than or equal to zero, while another bit combination may result in the branch being taken if some variable is less-than or equal to zero, and while yet another bit combination may result in the branch being taken if the previous arithmetic operation resulted in a negative result. Therefore, by setting the nullify bit, the displacement bit, and the three condition field bits, at least four discrete jump modes can be defined.

(ii) the execution of at least one subsequent instruction within said pipeline based on said assigned values of said first, second, and third data bits, when said at least one branch instruction is decoded. Again, from the table in the rejection of claim 34(b) above, it can be seen that a subsequent instruction's execution is dependent on the values of the aforementioned bits within the instruction word.

31. Referring to claim 42, Lee has taught a digital processor as described in claim 41. Lee has further taught that first and second of said at least four branch modes implement one- and two-cycle stalls within said pipeline, respectively. See column 5, lines 32-37, and Fig.2. Note that if a jump occurs and the displacement is positive (as shown in Fig.2, component 112), the delay slot instruction would be fetched but not executed. Therefore, in order to kill the unwanted instruction, the pipeline would be stalled for at least a single cycle. Also, see Fig.5, column 5, lines 58-63, and column 7, lines 21-25. Note that pipeline includes four stages: an address generation stage (A), a fetch stage (F), an execution stage (E), and a write stage (W). It has been

Art Unit: 2183

disclosed that the target address of the branch is not determined until the end of the execution stage. By this time, the delay slot instruction will have been fetched, and the predicted target address will be supplied to the program counter for fetching in the next cycle. See column 7, lines 6-10. If the delay slot instruction ends up being nullified (based on the displacement and nullify bits) and the predicted target is incorrect, then both instructions will need to be cancelled, resulting in a 2-cycle stall.

32. Referring to claim 43, Lee has taught a digital processor as described in claim 41. Lee has further taught that at least one of said at least four branch modes comprises a user-configurable mode. It should be noted that these branch instructions would have at least one user-configurable mode in that for every branch instruction, regardless of the mode, the programmer/user must specify the target address of the branch. This target will then determine what value is used for the displacement bit. In addition, the nullify bit is part of the instruction word. See column 3, lines 46-61. This nullify bit will be specified by the programmer/user in order to control the execution of the delay slot instruction.

Claim Rejections - 35 USC § 103

33. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

34. Claim 19 is rejected under 35 U.S.C. 103(a) as being unpatentable over Lee, as applied above.

Art Unit: 2183

35. Referring to claim 19, Lee has taught a processor as described in claim 17. Lee has further taught a 4-stage pipeline with an instruction address generation stage, an instruction fetch stage, an execute stage, and a write stage. See Fig. 5 and column 6, lines 56-64. Lee has not explicitly taught a stage just for decoding. However, Lee has taught that decoding is done in one of the aforementioned stages. See column 7, lines 12-16. Lee further states that the execution of instructions can be pipelined to any depth desired. See column 6, lines 63-64. It is well known in the art that pipelines include a separate decode stage. The actual implementation of the pipeline is a designer's preference but Lee has taught a system in which any size pipeline would suffice. Therefore, if desired, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement a pipeline with a separate decode stage, as is well known in the art.

36. Claims 26-28 and 30-32 are rejected under 35 U.S.C. 103(a) as being unpatentable over Lee, as applied to claims 1 and 14, respectively, in view of Kawasaki et al., U.S. Patent No. 5,530,965 (herein referred to as Kawasaki).

37. Referring to claims 26, 27, 30, and 31, Lee has taught a method and digital processor as described in claims 1, 25, 14, and 29, respectively.

a) Lee has further taught of providing an instruction word having an opcode and at least one short immediate value associated therewith, said at least one short immediate value comprising a plurality of bits. Note from column 3, lines 46-51, that the instruction format includes an 11 bit displacement field, which holds an immediate constant for branching purposes.

Art Unit: 2183

b) Lee has not explicitly taught selecting a portion of said plurality of bits of said at least one short immediate value, shifting all of said bits of said at least one short immediate value using said opcode and only said portion of bits to produce a shifted immediate value, and storing said shifted immediate value in a register. However, Kawasaki has taught such a concept for branching purposes as well as for other instructions, such as move instructions. See column 51, lines 50-55. Kawasaki has disclosed a move instruction of the format: **mov #imm, Rn**, where the short immediate value specified by #imm is sign-extended to form a long immediate value which is then stored in the register specified by Rn. By sign-extending an immediate value, a portion (the sign bit) of the value is copied into the most significant bit positions of the long immediate value. For instance, if the #imm field specified a short 4-bit immediate value 1010, which is to be transformed into an 8-bit long immediate value, then 1010 would be sign-extended to 11111010, where the sign bit of the 4-bit value is copied into the 4 most significant bit positions of the long value. It also follows that the 4-bit value has been shifted. Note that initially, 1010 contained a 1 in the most significant bit position, a 0 in the second most significant bit position, a 1 in the third most significant bit position, and a 0 in the fourth most significant bit position. After sign-extending the 4-bit value, the same numbers become the fifth, sixth, seventh, and eighth most significant bits, respectively. Hence, they have been shifted. Furthermore, in column 42, lines 16-27, Kawasaki has disclosed that this type of move instruction is used to help branch to addresses out of the short immediate value's range. More specifically, if a branch needs to branch further than what the short displacement allows, then the destination address is moved to the register specified by the "mov" instruction and a "jmp" instruction (shown in column 48, lines 28-30) is used with to jump to the address stored in the

Art Unit: 2183

register. A person of ordinary skill in the art would have recognized that this concept could be applicable in a system that is concerned with branching, such as Lee's. Such a concept would allow a branch instruction to branch to an address outside of the range of just a short immediate displacement value. This in turn would give a programmer more freedom in that they would not have to worry about program length or certain parts of a program being out of reach of a branch. Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to take Kawasaki's concept of selecting a portion of said plurality of bits of said at least one short immediate value, shifting all of said bits of said at least one short immediate value using said opcode and only said portion of bits to produce a shifted immediate value, and storing said shifted immediate value in a register, and apply it to the system of Lee.

38. Referring to claims 28 and 32, Lee in view of Kawasaki has taught a method and digital processor as described in claims 27 and 31, respectively. Recall that Lee has taught an instruction format that includes a 6-bit opcode field, meaning Lee's system has the ability of choosing between 64 different instructions if necessary. With the exception of branching, Lee has not explicitly stated any other instructions that have been implemented. However, it is inherent that other instructions would exist so that the processor can perform useful operations. A processor that does nothing but branching would do nothing useful. Furthermore, in column 3, lines 46-51, Lee has disclosed that his system is a register-register (load-store) architecture, i.e. where main memory is only accessed through load and store operations and operations are performed on values in registers. This is known because Lee has implemented an instruction format with two register operands. A move (mov) instruction is also well known in the art and is explicitly shown in Kawasaki. More specifically, Kawasaki has shown multiple versions of a

Art Unit: 2183

move instruction. One version moves a short immediate value into a register (as shown in column 51, lines 51 and 55-56) and another version includes moving a value from one register to another (as shown in column 52, line 53). A person of ordinary skill in the art would expect to find these common move instructions within the system of Lee because they allow a programmer to move an initial value into a register as well as temporarily store a register value into another register. Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to have at least one instruction word having a plurality of fields and said at least one instruction word having a short immediate value comprise the same instruction word(s); in this case, the move instruction, taught by Kawasaki.

Conclusion

39. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire **THREE MONTHS** from the mailing date of this action. In the event a first reply is filed within **TWO MONTHS** of the mailing date of this final action and the advisory action is not mailed until after the end of the **THREE-MONTH** shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than **SIX MONTHS** from the date of this final action.

Art Unit: 2183

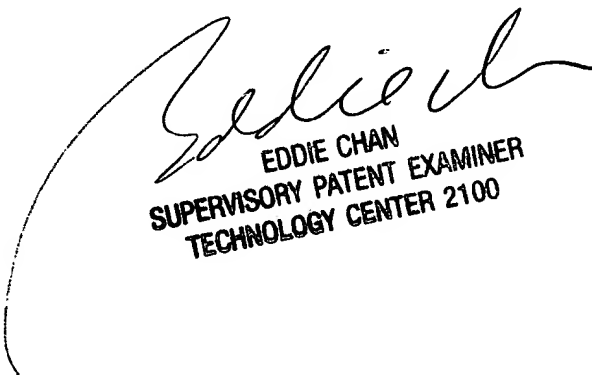
Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (703) 305-7811.

The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (703) 305-9712. The fax phone numbers for the organization where this application or proceeding is assigned are (703) 746-7239 for regular communications and (703) 746-7238 for After Final communications.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

DJH
David J. Huisman
May 15, 2003



EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100